# **Final Report**
for $100K supplement to ONR funded research on
Event Representation in Humans and Machines

Joseph Bates
Visiting Scientist
Cognitive Machines Group
MIT Media Lab

Background and Objective

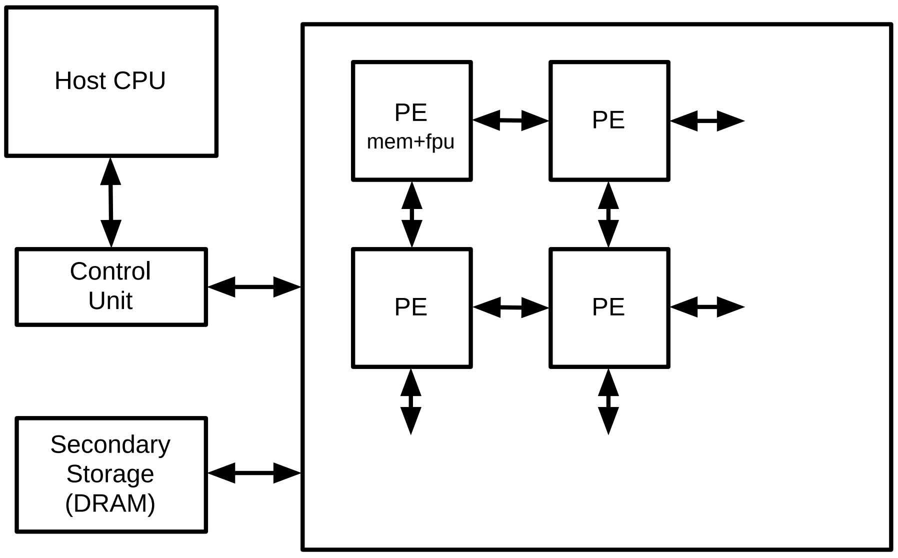
ONR has been funding Prof. Roy and Prof. Baldwin to perform research on *Event Representation in Humans and Machines.* Prof. Roy's portion of that research involves developing an unsupervised learning system that acquires structured event representations of human activity grounded in naturalistic video observations. The output of that system is to be an action lexicon that encodes recurrent hierarchical temporal patterns discovered from a large video corpus.

The initial video corpus contains 90,000 hours of video. One of the most compute intensive tasks in analyzing such large video databases, both for learning the action lexicon and for subsequently recognizing instances of the lexical elements, is robust tracking of objects and people. Existing hardware and software limit the efficiency with which tracking and other key analyses can be performed.

The goal of the work undertaken with the supplemental funding was to determine whether Dr. Bates's proposed "approximate computing" technology could radically improve the speed and energy consumption of the tracking task.

Bates's technology is built on the idea of a massively parallel programmable SIMD coprocessor, fitting roughly 100,000 processing elements (PEs) on a single chip, where each PE is capable of rapidly performing very low accuracy floating point arithmetic (approximately 1% error per operation). Each PE consists of approximately ten thousand transistors and contains, besides the arithmetic circuits, roughly 100 words of local memory and a somewhat flexible local interconnect scheme. For well-suited tasks, where software can recover sufficiently accurate results using the relatively inaccurate underlying hardware, the machine is able to compute thousands of times mores efficiently than CPUs and hundreds of times more efficiently than GPUs and FPGAs.

The following is a diagram showing the basic architecture of the processor and coprocessor system:



Research Activities Undertaken

In order to determine whether the Cognitive Machines Group's tracking technology could be accelerated, Dr. Bates worked with George Shaw, a Master's student in the group, to understand the group's existing technology and to attempt to adapt it to run on Bates's proposed massively parallel, approximate arithmetic, SIMD hardware.

Mr. Shaw was developing a tracking system that combined a variety of tracking methods into an overall tracker, with the expectation that overall tracking performance would exceed the performance of the individual trackers. The overall tracker was structured as a pipeline. The first step in the supplemental research was to understand what tasks in this pipeline were slow and would most benefit from acceleration. Below is a brief overview of the pipeline, followed by discussion of the speed and feasibility of acceleration for relevant stages.

The first stage is reading video from disk. Some of the research undertaken by the Cognitive Machines Group using the video library requires the frames to be randomly accessible. For this reason, the video is compressed as a collection of independent frames using JPEG image compression. This is in contrast to more effective time-aware compression, such as MPEG, which is not used because such compression does not easily admit fast random access to frames. So the first step in the analytics pipeline is to read the JPEG frames from disk and decode them.

Once a frame is decoded and available, it is analyzed to distinguish foreground pixels from background pixels. Background pixels are intended to be the (approximately) unchanging part of the video, while foreground pixels are the components that make up the objects to be tracked.

Multiple trackers operate on the foreground pixels. A motion tracker groups the foreground pixels into connected components, then attempts to track motion of those components. A color-based tracker compares the color distribution of foreground pixels from frame to frame, forms a probability density map based on the color correspondence, and then tracks the high density regions using a variant of the mean-shift algorithm.

Higher levels combine the results of the multiple trackers, handle disappearance and coalescing of objects, and produce a final set of tracking results.

Mr. Shaw reported that the first stages of the pipeline, reading and decompressing frames and foreground/background separation, accounted by a large margin for the bulk of the processing time. So we decided to focus on determining whether those tasks could be implemented and accelerated using the low precision SIMD architecture.

Foreground/Background Separation

The first task we looked at was foreground/background separation. This was being done using a "MOG" approach - where a mixture of Gaussians model is attached to each pixel position in the video. The model specifies a set of Gaussians, each defined by a mean and a standard deviation. The values are measured in some color space component, such as luminosity. As frames are processed, each pixel is compared to its model. If, on a given frame, a pixel's value falls within a specified distance (measured in standard deviations) from the mean of one of the Gaussians, then the pixel is considered to be background. Otherwise, it is considered foreground. Further, the models are modified on each frame so that modest deviations from an existing mean will cause the corresponding Gaussian to migrate toward the new pixel value, while new Gaussians will be created (and not recently matched Gaussians will be deleted) when there are sufficiently many occurrences of pixel values not matching any existing Gaussian.

This well known method works fairly well for foreground/background separation. It has the disadvantage that a temporarily stable foreground object gradually will fade to become background. It also has the disadvantage of requiring substantial floating point computing at every pixel of every frame. However, this computing pattern maps well to a SIMD architecture, because if, say, each pixel position is assigned to a core (Processing Element), and if the new frame is loaded into the SIMD grid, then all cores can compare their locally stored pixel with their locally stored models and perform the steps of the MOG algorithm with relatively little waste of computing capacity. (The compute time is proportional to the maximum number of models stored at any pixel position, which is a small constant number, such as typically 5 in Shaw's explorations.)

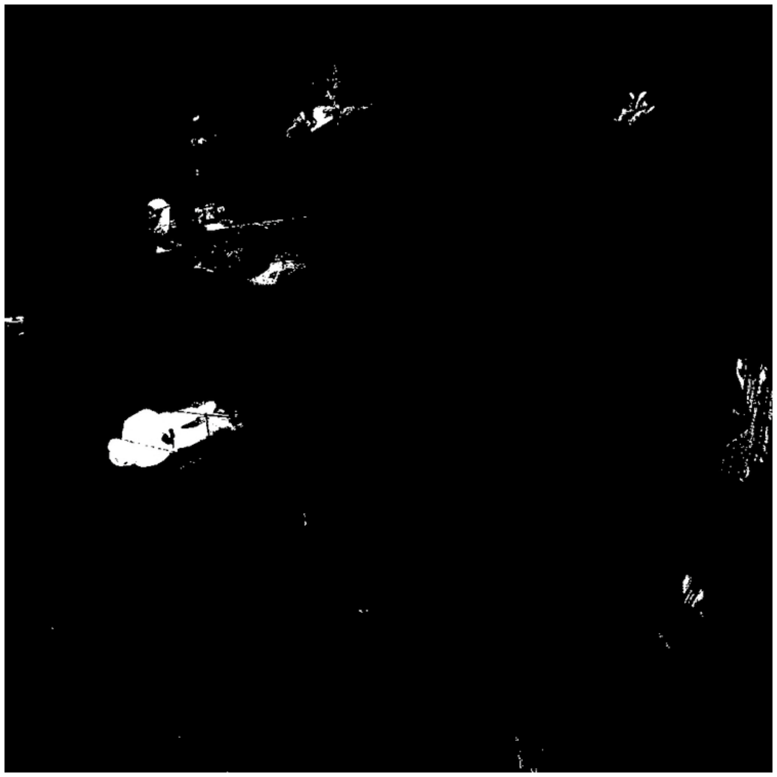
Thus, MOG should make efficient use of the SIMD hardware, and the time taken to run MOG would be some thousands of machine instructions per frame. This means processing times in the tens of microseconds per frame, once the frame is loaded into SIMD memory. The implementation of MOG that the group currently runs on serial machines takes tens of milliseconds per frame.

The question about this approach is whether doing the Gaussian-related computations in approximate floating point produces sufficiently good results. That is a question that is difficult to analyze theoretically, but easy to test experimentally. So we undertook such an experiment.

An advantage of Bates's architecture is that its approximate arithmetic can be exactly modeled using a traditional digital computer with only about 10x overhead. Bates provided a set of routines that precisely emulated the primitive hardware arithmetic - Add, Subtract, Multiply, Divide, Sqrt. On top of these Bates and Shaw built routines needed to do the Gaussian modeling, such as exponentiation. The latter, for instance, was expressed as several leading terms of the Taylor expansion of e^-x.

Shaw's existing MOG code was then modified to run using low precision arithmetic and to produce exactly the same results that would be produced by an actual hardware implementation. The code was run on a two minute sample of video extracted from the group's video library. It was compared to the same algorithm run using full floating point arithmetic on a traditional CPU.

An example frame, processed using traditional floating point, is below. The same frame, reprocessed using Bates's proposed low precision arithmetic, follows.



Floating point foreground/background separation

Low precision foreground/background separation

Displayed side by side, to the eye, these images appear identical. Over the complete two minute sequence, the average number of classification differences per frame was .0014% (ie, 14 pixels differed on average in each 1M pixel frame).

JPEG decompression

With good results on foreground/background separation, we turned out attention to a paper study of JPEG decompression. The general method of JPEG encoding, as described on Wikipedia, is as follows:

- The representation of the colors in the image is converted from RGB to Y′CBCR, consisting of one luma component (Y'), representing brightness, and two chroma components, (CB and CR), representing color. This step is sometimes skipped.

- The resolution of the chroma data is reduced, usually by a factor of 2. This reflects the fact that the eye is less sensitive to fine color details than to fine brightness details.

- The image is split into blocks of 8×8 pixels, and for each block, each of the Y, CB, and CR data undergoes a discrete cosine transform (DCT). A DCT is similar to a Fourier transform in the sense that it produces a kind of spatial frequency spectrum.

- The amplitudes of the frequency components are quantized. Human vision is much more sensitive to small variations in color or brightness over large areas than to the strength of high-frequency brightness variations. Therefore, the magnitudes of the high-frequency components are stored with a lower accuracy than the low-frequency components. The quality setting of the encoder (for example 50 or 95 on a scale of 0–100 in the Independent JPEG Group's library[15]) affects to what extent the resolution of each frequency component is reduced. If an excessively low quality setting is used, the high-frequency components are discarded altogether.

- The resulting data for all 8×8 blocks is further compressed with a lossless algorithm, a variant of Huffman encoding.

JPEG decoding performs the above steps in reverse order, though step 4, the quantization step, produces loss of information when reversed.

The most computationally intense step of the decoding process is the inverse discrete cosine transform. Whereas the other steps take a modest amount of computing per pixel, the DCT requires a 64 element (8x8 pixel) convolution of floating point numbers for every pixel. However, the convolution is local and the weights of each component of the convolution are the same everywhere in the image, so if the image is loaded into the processing elements of the 2D hardware array, each weight can be sequentially broadcast to all processing elements simultaneously, and the convolutions can be performed across the whole image in parallel.

JPEG decoding, in general, requires high precision. For instance, some methods of encoding video, using JPEG as a component, require that the encoder know precisely what the decoder will do. However, for the application we are considering, where each frame is encoded independently and where we want to do tracking, which is a method that must be (and is) robust to noise in its inputs, such as, for instance, when performing foreground/background separation, we do not expect that high precision will be needed. Compressing and expanding sample images using JPEG shows that with reasonable compression ratios pixel values frequently vary from source to reconstruction by 5%. From other experience using the low precision approach, we believe errors due to low precision arithmetic should be below this level and should not substantially degrade the JPEG decompression results.

Besides the parallelism inherent in performing DCT at every pixel in an image, JPEG encodings are composed of a series of encoded data blocks, with each chunk of blocks terminated by a “restart marker.” The encoder resets at each restart marker, which means the chunks can be decoded independently. This means that a JPEG file can be split into a collection of entirely independent subfiles, each of which can be decoded in parallel. There is much SIMD-style parallelism inherent in JPEG decompression.

JPEG reasonably achieves compression ratios of 100x. Loading an image into the SIMD machine in compressed form, decompressing it in place, then continuing the analytics pipeline (such as foreground/background separation) without needing to move the uncompressed image between CPU and accelerator, allows us to get an effective 100x improvement in bandwidth of the CPU/accelerator bus. This bypasses one of the main places where Amdahl’s law could limit acceleration - sending image data from disk to CPU to accelerator.

Our conclusion in analyzing JPEG decompression is that it is likely that the process can be performed on a low precision SIMD machine, yielding sufficiently good results to enable subsequent stages of the analytics pipeline to function, while making efficient use of the hardware and thus achieving great speedup and low energy consumption.

Feature based tracking

Feature-based tracking is another well known and powerful tracking method, but it has been too slow for Speechome to consider. Subsequent to the work described here, in commercial work funded by an ONR SBIR award, Bates has been assisting a group of vision researchers study feature-based tracking using SIFT-like features. Tracking algorithms run using the emulated arithmetic work well in low precision.

Conclusion

Accelerating tracking using massively parallel, low arithmetic precision, SIMD architectures continues to look very promising. Foreground/background separation works surprisingly well. Getting around bus bandwidth bottlenecks by doing JPEG decompression on the accelerator appears likely to work well. In subsequent independently performed work, feature-based tracking has been shown to work well.

Ongoing analysis of the underlying hardware continues to show that there are no conceptual difficulties requiring further research, and that the hardware is manufacturable at extremely low cost per arithmetic operation per second. Tracking looks feasible at very low energies, such as required in small UAVs, at very low cost, such as required in mass market consumer products, and at the enormous speeds and low costs required to accelerate Speechome-like video analytics in a research environment.

Thus, we believe the answer to our original question, whether Dr. Bates's proposed "approximate computing" technology could radically improve the speed and energy consumption of the tracking task, is yes.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 30-09-2011	2. REPORT TYPE Final Technical Report	3. DATES COVERED *(From - To)* 04/15/2010 - 04/14/2011

4. TITLE AND SUBTITLE Event Representation in Humans and Machines	5a. CONTRACT NUMBER
	5b. GRANT NUMBER N00014-10-1-0829
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Roy, Deb Bates, Joseph	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Massachusetts Institute of Technology 77 Massachusetts Ave. Cambridge, MA 02139	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Bello, Paul (Technical Representative) Office of Naval Research 875 North Randolph Street; ONR 341 Arlington, Virginia	10. SPONSOR/MONITOR'S ACRONYM(S) 000146 - Navy - ONR
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for Public Release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

One of the most compute intensive tasks in analyzing naturalistic video is tracking objects and people. Tracking complete databases containing hundreds of thousands of hours of video has traditionally been extremely time consuming and/or expensive. The massively parallel, low arithmetic precision, SIMD architecture proposed by Bates was studied to determine whether it could bring great efficiency benefits to tracking. The slowest subtasks in the tracking pipeline were studied, and it appears that tracking is a task that maps well to the proposed hardware, with the potential for thousands of times speedup, lower energy use, and cost, compared to traditional CPU-based methods.

15. SUBJECT TERMS

analyzing naturalistic video, tracking pipeline, massively parallel, low arithmetic precision, SIMD architecture

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Roy, Deb
U	U	U	SAR	7	19b. TELEPHONE NUMBER *(Include area code)* 617-253-0596